Một bổ sung mới quan trọng cho SDL 2 là API (Giao diện chương trình ứng dụng - Application Programming Interface) render kết cấu. Điều này mang lại cho ta render dựa trên phần cứng nhanh và linh hoạt. Trong phần hướng dẫn này, chúng ta sẽ sử dụng kỹ thuật render mới.

|  |
| --- |
| // Loads individual image as texture  SDL\_Texture \*loadTexture(std::string path);  // The window we'll be rendering to  SDL\_Window \*gWindow = NULL;  // The window renderer  SDL\_Renderer \*gRenderer = NULL;  // Current displayed texture  SDL\_Texture \*gTexture = NULL; |

* Kết cấu (texture) trong SDL có 1 loại dữ liệu riêng là SDL\_Texture. Khi ta phải đối mặt với SDL Texture, ta cần một SDL\_Renderer để render nó vào màn hình, đó là lý do chúng ta khai báo một biến toàn cục “ gRenderer”.
* Bạn cũng có thể thấy chúng tôi có một thói quen load ảnh với loadTexture và một texture được khai báo toàn cục mà chúng tôi định load.

|  |
| --- |
| // Create window          gWindow = SDL\_CreateWindow("SDL Tutorial", SDL\_WINDOWPOS\_UNDEFINED, SDL\_WINDOWPOS\_UNDEFINED, SCREEN\_WIDTH, SCREEN\_HEIGHT, SDL\_WINDOW\_SHOWN);          if (gWindow == NULL)          {              printf("Window could not be created! SDL Error: %s\n", SDL\_GetError());              success = false;          }          else          {  **// Create renderer for window**  **gRenderer = SDL\_CreateRenderer(gWindow, -1, SDL\_RENDERER\_ACCELERATED);**  **if (gRenderer == NULL)**  **{**  **printf("Renderer could not be created! SDL Error: %s\n", SDL\_GetError());**  **success = false;**  **}**  **else**  **{**  **// Initialize renderer color**  **SDL\_SetRenderDrawColor(gRenderer, 0xFF, 0xFF, 0xFF, 0xFF);**  **// Initialize PNG loading**  **int imgFlags = IMG\_INIT\_PNG;**  **if (!(IMG\_Init(imgFlags) & imgFlags))**  **{**  **printf("SDL\_image could not initialize! SDL\_image Error: %s\n", IMG\_GetError());**  **success = false;**  **}**  **}**          } |

* Sau khi tạo cửa sổ, chúng ta có thể tạo một renderer cho window của chúng ta, vì thế, chúng ta có thể render texture lên nó. May thay điều này khá dễ để thực hiện với một lệnh **SDL\_CreateRenderer()**.
* Sau khi tạo renderer, chúng ta muốn khởi tạo một màu render bằng **SDL\_SetRenderDrawColor()**. Điều này kiểm soát được màu nào được dùng cho những toán tử (operation) render khác nhau.

|  |
| --- |
| **SDL\_Texture \*loadTexture(std::string path)**  **{**  **// The final texture**  **SDL\_Texture \*newTexture = NULL;**  **// Load image at specified path**  **SDL\_Surface \*loadedSurface = IMG\_Load(path.c\_str());**  **if (loadedSurface == NULL)**  **{**  **printf("Unable to load image %s! SDL\_image Error: %s\n", path.c\_str(), IMG\_GetError());**  **}**  **else**  **{**  **// Create texture from surface pixels**  **newTexture = SDL\_CreateTextureFromSurface(gRenderer, loadedSurface);**  **if (newTexture == NULL)**  **{**  **printf("Unable to create texture from %s! SDL Error: %s\n", path.c\_str(), SDL\_GetError());**  **}**  **// Get rid of old loaded surface**  **SDL\_FreeSurface(loadedSurface);**  **}**  **return newTexture;**  **}** |

* Hàm load texture của chúng ta vẫn nhìn phần lớn như trước, chỉ là thay vì chuyển surface được load sang định dạng display, thì ta tạo một texture từ surface được load sử dụng **SDL\_CreateTextureFromSurface()**. Giống như trước đây, hàm này sẽ tạo một texture mới từ một surface đã có, nghĩa là cứ như trước đây ta phải giải phóng surface được load và sau đó trả về texture được load.

|  |
| --- |
| **bool loadMedia()**  **{**  **// Loading success flag**  **bool success = true;**  **// Load PNG texture**  **gTexture = loadTexture("07\_texture\_loading\_and\_rendering/texture.png");**  **if (gTexture == NULL)**  **{**  **printf("Failed to load texture image!\n");**  **success = false;**  **}**  **return success;**  **}**  **void close()**  **{**  **// Free loaded image**  **SDL\_DestroyTexture(gTexture);**  **gTexture = NULL;**  **// Destroy window**  **SDL\_DestroyRenderer(gRenderer);**  **SDL\_DestroyWindow(gWindow);**  **gWindow = NULL;**  **gRenderer = NULL;**  **// Quit SDL subsystems**  **IMG\_Quit();**  **SDL\_Quit();**  **}** |

* Vì load texture trừu tượng cới hàm load ảnh của chúng ta, hàm **bool loadMedia()** vẫn hoạt động như cũ
* Trong hàm dọn, chúng ta phải nhớ hủy cái texture dùng **SDL\_DestroyTexture()**.

|  |
| --- |
| **// While application is running**  **while (!quit)**  **{**  **// Handle events on queue**  **while (SDL\_PollEvent(&e) != 0)**  **{**  **// User requests quit**  **if (e.type == SDL\_QUIT)**  **{**  **quit = true;**  **}**  **}**  **// Clear screen**  **SDL\_RenderClear(gRenderer);**  **// Render texture to screen**  **SDL\_RenderCopy(gRenderer, gTexture, NULL, NULL);**  **// Update screen**  **SDL\_RenderPresent(gRenderer);**  **}** |

* Trong main loop sau event loop, chúng ta gọi **SDL\_RenderClear()**. Hàm này sẽ để màn hình có máu cuối cùng ta set với SDL\_SetRenderDrawColor().
* Với màn hình đã được clear, chúng ta render texture với **SDL\_RenderCopy()**. Với texture render được. Chúng ta vẫn phải cập nhật màn hình, nhưng vì chúng ta không dùng SDL\_Surface nên chúng ta không thể dùng **SDL\_UpdateWindowSurface()**. Thay vào đó, chúng ta phải dùng **SDL\_RenderPresent()**.
* Bây giờ có một API mới là IMG\_LoadTexture. Nó không phải trong document chính thức kể từ bài viết này nhưng bạn có thể tìm nó trong file SDL\_image header. Nó cho phép bạn load một texture mà không phải tạo một surface tạm thời. Lý do tôi trở lại và update cái tutorial để sử dụng nhưng phương pháp này vì tôi không muốn trở lại 40+ tutorials.
* Hãy thử nó. Đồng thời hãy làm quen với việc phải dùng code không đc hướng dẫn vì thường thì thời gian trong phát triển game chuyên nghiệp, có tài liệu thật sang trọng.